Program 3 Project Report Samuel Gonzalez Checa

Colab Program URL:

https://colab.research.google.com/drive/1SyOA85rqTbPiVrBN7ur_PGv7T-qNs-P0?usp=sharing

Github Repo URL: https://github.com/SamGonChec/TensorFlow_Predict_Job_Change

Intro:

I began by encoding the data through get_dummies(). I encoded my data in a separate file (URL: https://github.com/SamGonChec/TensorFlow_Predict_Job_Change/blob/main/src/program_3_samuel_gonzalez_checa_encoding.py)

I then decided to take in that encoded data and normalize the columns: experience, last_new_job, and training_hours. After attempting with many modifications, I have a model that reaches a decent level of accuracy.

Model:

I divided the model as instructed with a 70/15/15. I shuffled and split the data using the sample method from pandas. After I set the target column to a specific variable to check each of my subset of data. At this point I was having issues with the training since the target data and the target to compare it to, these two variables had different dimensions. I then used keras.utils.to_categorical to encode the data. This would turn the target data into a numpy binary matrix. This made it compatible with the data and I was able to compare the two.

I used Keras Sequential model and added two hidden layers. The first hidden layer has 100 nodes and it uses the activation of tanh. The second one has 50 nodes and uses the same activation. In the outer layer I allow for two outputs and a softmax activation.

I compiled the model using the compile method. I used the Adam optimizer and mean_squared_error loss function. I then trained the model using the fit method that takes in the data and the target data and compares them. This fit function also takes the validation data and checks it after each epoch. The number of Epochs I used were 50, and the batch size was 10.

This model has reported a training of: 80.72%

Validation of: 78.04%

Testing of: 77.87%

Attempts at improving the Testing and Training:

There were multiple things I attempted to figure out why I was getting low testing percentages. I first began by ignoring the whole encoded_data.csv and just get_dummies (one-hot coded) everything in the original aug_train.csv file. I had around 70 columns instead of 45 like my encoded_data file. Unfortunately this did not help at all.

I then decided to change the loss function in the compile method of the model. I tried using the categorical crossentropy but this did not help. I also tried using another optimizer like Nadam or Adamax, but still no improvement. I ended up trying anything to see if it stuck in the Keras documentation, but nothing: https://keras.io/api/optimizers/. I lastly decided to change the activation formulas and the number of output nodes, and this threw the model off. It seemed to have corrupted the training since I would only get 50% throughout the 50 epochs. This made me return to the previous activation and output nodes (2 nodes and softmax) since single and sigmoid did not help.

In the layers I also tried having more nodes, having less nodes. I even tried to drop off data during the training but this did not help. Rather when I dropped off the data the percentages dipped by double digits.

Conclusion:

I came to the conclusion throughout this whole project that I might be mishandling the data or that there must be something wrong with the data in the first place. I tried everything I had access to and nothing worked. The model seems to be training since I am evaluating the data before and after the training and there is an improvement. Before the training my testing and validating were around 69%, but after training it caps at 78%. Which makes me think that there is something wrong with the data and that it must be a cap, training the model further only overfits the network.